PRIMA
Volume 17, Nomor 1, Juni 2020 **ISSN : 1411-0296**

RANCANGAN PERANGKAT LUNAK AKUISISI DATA MODUL DETEKTOR GAMMA RosRao BERBASIS *MODBUS OVER TCP/IP* MENGGUNAKAN *PyQT5*

Mohamad Amin, Joko Triyanto, Istofa
Pusat Rekayasa Fasilitas Nuklir – BATAN
Gedung 71 Kawasan PUSPIPTEK Serpong, Tangerang Selatan 15314
amin123@batan.go.id ; triyanto123@gmail.com

ABSTRAK

RANCANGAN PERANGKAT LUNAK AKUISISI DATA MODUL DETEKTOR GAMMA BERBASIS PROTOKOL MODBUS TCP/IP MENGGUNAKAN PyQT5. *Paper ini membahas rancangan perangkat lunak akuisi data pada modul detektor gamma buatan RosRoa Rusia. Modul detektor tersebut menggunakan protokol modbus RTU yang dilewatkan pada protokol jaringan berbasis TCP/IP. Rancangan ini bertujuan untuk menguji format dan jenis perintah yang dikirim ke modul deteksi dan untuk menganlisis data yang dikirim kembali oleh modul tersebut ke komputer master. Rancangan menggunakan toolkit PyQT5 yang menjadi perantara antara bahasa pemrograman python dan Graphic User Interface (GUI) dari QT. Hasil rancangan adalah sebuah perangkat lunak aplikasi yang dapat mengirim perintah ke modul detektor dan menerima respon balik melalui protokol modbus yang dilewatkan pada protokol TCP/IP.*

Kata kunci : Rancangan, akuisisi data, Modbus, TCP/IP, PyQT 5.

ABSTRACT

A DESIGN OF DATA ACQUISITION SOFTWARES FOR GAMMA DETECTOR MODULE BASED ON MODBUS OVER TCP / IP PROTOCOL USING PYQT5. *This paper discusses a design of the acquisition softwares on a gamma detector module made by RosRoa Russia. The detector module uses the modbus RTU protocol which is passed over TCP / IP-based network protocols. The design aims to test the format and type of commands sent to the detection module and to analyze the data sent back by the module to the master computer. The design uses the PyQT5 toolkit which interfaces between the python programming language and the Graphic User Interface (GUI) of QT. The result of the design is an application software that can send commands to the detector module and receive a response back through the modbus protocol which is passed to the TCP / IP protocol.*

Keywords: Design, data acquisition, Modbus, TCP / IP, PyQT 5.

1. PENDAHULUAN

Sistem deteksi radiasi gamma yang digunakan dalam perekayasaan instrumentasi nuklir di Pusat Rekayasa Fasilitas Nuklir Batan (PRFN_BATAN) umumnya dilakukan dengan mengintegrasikan beberapa modul. Sejumlah modul dibuat dan dirakit sendiri, sementara modul deteksi diperoleh dengan cara membeli dari vendor instrumentasi nuklir yang berbeda-beda di luar negeri. Cara pengadaan seperti ini belum terhindarkan karena pihak produsen dalam negeri belum ada yang mampu membuat produk-produk instrumentasi nuklir serupa. Akibatnya, modul yang diperoleh dari pengadaan luar negeri umumnya masih memerlukan sejumlah modifikasi atau penambahan modul antarmuka agar dapat diintegrasikan dengan modul-modul yang dibuat atau dirakit sendiri.

Sejumlah permasalahan yang sering dijumpai ketika pengadaan modul sistem deteksi yang berasal dari penyadia luar negeri adalah tidak tersedianya dokumen pengembangan yang disertai "*source-code*" dari suatu bahasa pemrograman, bentuk

protokol komunikasi yang digunakan antara modul satu dan lainnya berbeda, software aplikasi bawaan untuk mengakuisisi data dari modul menggunakan bahasa asing dan belum dapat terkoneksi langsung ke sistem yang sedang dikembangkan. Pendek kata, modul sistem deteksi radiasi gamma yang dibeli dari luar masih memerlukan modifikasi atau antarmuka sebelum diintegrasikan dengan modul atau sistem instrumentasi yang sedang dikembangkan.

Permasalahan yang sama terjadi pada modul sistem deteksi monitor lingkungan buatan perusahaan instrumentasi nuklir Rusia. Permasalahan tersebut antara lain, modul sistem deteksi tidak didukung oleh dokumen pengembangan yang memadai sehingga format perintah yang harus dikirim dan jenis data yang diterima belum dapat diketahui secara rinci. Permasalahan lain adalah penggunaan protokol komunikasi pertukaran data menggunakan protokol *modbus TCU (Remote Terminal Unit)* yang dilewatkan pada protokol komunikasi jaringan berbasis *TCP/IP* (*Transmission Control Ptotocol*/*Internet Protocol)*. Permasalahan ini membutuhkan suatu perangkat lunak akuisisi data agar dapat diintegrasikan ke modul yang telah dibuat. Disamping itu, perangkat lunak aplikasi yang disertakan pada saat pembelian tidak dirancang untuk dapat diintegrasikan dengan modul atau sistem perangkat lunak monitor lingkungan yang sedang dikembangkan. Untuk mengatasi permasalahan ini, sebagai anggota dari tim perekayasaan monitor lingkungan, kami mencoba merancang perangkat lunak akuisisi data untuk menguji dan atau menganalisis perintah-perintah yang dikirim ke modul atau sistem deteksi, maupun juga jenis data yang dikirim balik dari sistem deteksi tersebut.

Paper ini mencoba menyajikan rancangan program aplikasi akuisisi data menggunakan *toolkit PyQT5*. Maksud pembuatan rancangan adalah untuk menguji bentuk perintah yang dikirim ke modul deteksi gamma menggunakan protokol *modbus RTU* yang dilewatkan pada protokol *TCP/IP*, dan menganalisis bentuk data yang diterima dari sistem deteksi.

2. DASAR TEORI

2.1. Protokol *Modbus*

Mod bus adalah protokol komunikasi pertukaran data yang dipublikasikan pertama kali oleh Modicon pada tahun 1979[1]. *Modbus* merupakan aplikasi yang berada pada lapisan teratas (lapisan ke-7) dari model arsitektur komunikasi *Open System Interconnection (OSI)*. Untuk melakukan pertukaran data antara sebuah komputer yang bertindak sebagai *master* dan sebuah sensor yang bertindak sebagi *slave, modbus* membutuhkan sebuah antarmuka dari perangkat keras fisik. Salah satu perangkat fisik tersebut adalah kartu jaringan yang disebut *Ethernet*[2,3,4]. Agar pertukaran data oleh *modbus* dapat dilakukan melalui kartu ethernet, *modbus* memerlukan sebuah protokol jaringan. Salah satu protokol jaringan tersebut adalah *TCP/IP*. Representasi format perintah *request* dari *master* dipecah ke dalam kelompok frame seperti diperlihatkan dalam Tabel 1[2,3,4,5].

Tabel 1. Format Perintah Request dari *Master*

Kode Alamat Slave	Kode Fungsi	Register	Panjang Data	CRC_16
1 byte	1 byte	2 byte	2 byte	2 byte

Format perintah *master* (Tabel 1) yang terdiri dari 5 memiliki panjang 8 byte. *Frame* pertama berisi kode dari alamat *slave.* Kode alamat *slave* bersifat unique, yaitu setiap sensor yang terhubung sebagai *slave* memiliki alamat yang berbeda. Ukuran kode alamat adalah 1 byte, sehingga jumlah sensor yang dapat terhubung pada protokol

modbus maksimal 255. Paper ini dua sensor, yaitu sebuah detektor *CsI(TI)* dengan alamat 1, dan sebuah detektor *Geiger Muller* dengan alamat 03.

Frame berisi kode fungsi perintah *request* dari *master*. Kode perintah *request* antara lain adalah perintah baca, perintah tulis dan perintah preset. Paper ini menggunakan 3 jenis kode perintah. Kode 0x03 untuk membaca register *holding*, kode 0x06 untuk melakukan preset satu *register holding*, dan kode 0x10 untuk melakukan preset ke sejumlah *register input* di *slave*.

Frame ketiga berhubungan dengan alamat dari *register* pertama yang akan dibaca atau diberi nilai. Misal, bila frame tersebut bernilai 5, *register* pertama di *slave* yang akan dibaca atau diberi nilai adalah *register* ke 6.

Frame keempat adalah frame panjang data. Frame panjang data berhubungan dengan jumlah register yang akan dibaca nilainya. Bila frame panjang data bernilai 3, nilai ini mengindikasikan jumlah register yang akan dibaca berjumlah tiga, terhitung mulai dari register pertama. Jadi, jika register pertama dimulai dari alamat 6, berarti register sisanya adalah register 7 dan 8.

Frame ke lima merupakan frame yang digunakan untuk memeriksa konsistensi format perintah yang dikirim. Frame ini akan dibaca oleh slave. Jika hasil pemeriksaan CRC_16 yang terkirim dan CRC_16 hasil validasi tidak sesuai, maka slave akan menolak perintah tersebut dan meminta ke *master* untuk melakukan pengiriman ulang perintah. CRC_16 yang digunakan dalam perancangan ini menggunakan persamaan polinomial 0xA001[4,5].

Untuk memastikan bahwa slave telah memproses perintah permintaan *master*, slave akan mengirim respon ke *master* berupa resume perintah yang diikuti sejumlah data. Representasi format respon *modbus* untuk register *holding* yang berasal dari slave dikelompokkan ke dalam 5 frame sebagai berikut (Tabel 2):

Tabel 2. Format Respon dari Slave

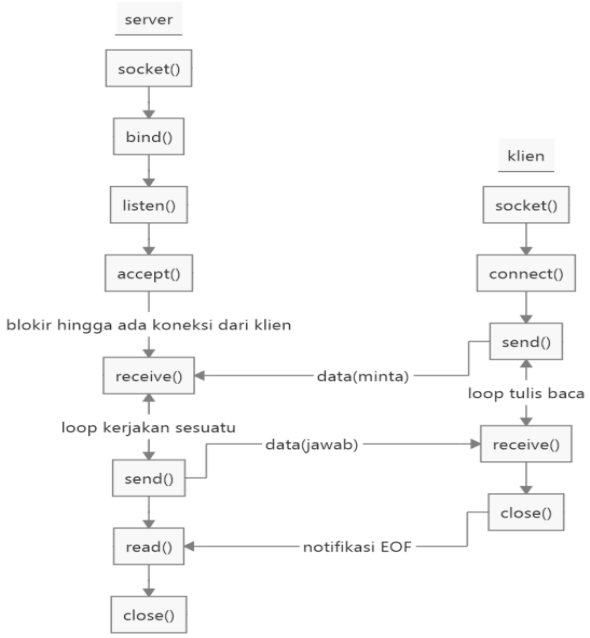
Kode Alamat Slave	Kode Fungsi	Jumlah byte resppon	Jumlal byte data tiap register	CRC_16
1 byte	1 byte	2 byte	2 byte	2 byte

2.2. Protokol *TCP/IP*

TCP/IP adalah protokol gabungan antara *TCP* (*Transmission Control Protocol*) dan *IP* (*Internet Protocol*)[6,7,8,9]. Kedua protokol ini mengatur komunikasi data dalam suatu proses pertukaran data antara komputer klien dan komputer server di dalam jaringan internet dan memastikan bahwa pengiriman data sampai ke alamat yang dituju. *TCP* berfungsi untuk melakukan proses koneksi, sementara *IP* bertugas memberikan pelabelan atau penomoran terhadap komputer yang akan menjadi tujuan dalam komunikasi pertukaran data.

Komunikasi pertukaran data antara komputer di dalam suatu jaringan dengan protokol *TCP/IP* memerlukan antarmuka pemrograman aplikasi (*Aplicaton Programming Interface*). Salah satu antarmuka tersebut dikenal dengan sebutan pemrograman *socket*[9]. Rancangan perangkat lunak akuisisi data modul detektor gamma *RosRoa* menggunakan pemrograman *socket*.

Skema rancangan pemoragraman berbasis *socket* adalah skema berorientasi koneksi (*connection oriented*). Dengan skema ini, pertukaran data antara komputer (*master*) dan sensor (*slave*) baru dapat dilakukan ketika keduanya ttelah terkoneksi. Urutan proses koneksi dan pertukaran data dengan skema berorientasi koneksi diilustrasikan dalam Gambar 1.



Gambar 1. Skema pertukaran data model orientasi koneksi *TCP/IP*.

Langkah-langkah untuk membangun komunikasi pertukaran data menggunakan perantara *socket TCP* dalam Gambar 1, dapat dijelaskan sebagai berikut:

A. Langkah-langkah pada sisi komputer yang memberikan respon (server):
- Pertama adalah *server* membuat *socket* dengan fungsi *socket()*
- Kedua, *socket* diikat ke alamat suatu alamat IP dan Port menggunakan fungsi *bind()*
- Ketiga, *server* mendengarkan koneksi dengan fungsi *listen()* ;
- Keempat, *server* menerima koneksi dengan panggilan sistem fungsi *accept*() . Panggilan ini umumnya diblokir sampai klien terhubung dengan *server*.
- Berikutnya, *server* menerima dan mengirim data dengan menggunakan fungsi *receive*() dan *send()* .
- Terakhir, *server* menutup koneksi dengan menggunakan fungsi close ().

B. Langkah-langkah pada sisi komputer yang melakukan permintaan (klien):
- Langkah kesatu adalah klien membuat soket menggunakan fungsi *socket().*
- Kedua, klien melakukan koneksi soket ke alamat server dengan fungsi *connect().*
- Berikutnya, klienmenggunakan fungsi kirim dan terima data melalui fungsi *receive()* dan fungsi *send()* .
- Tutup koneksi dengan menggunakan fungsi *close().*

2.3. *PyQT5*

PyQT5 adalah sebuah toolkit wijet *Graphic User Interface* (GUI), yaitu sebagai antarmuka grafik antara bahasa pemrograman *python* dan *QT* [10,11,12]. *PyQT* menyediakan sejumlah pustaka yang dapat digunakan oleh bahasa pemrgoraman *python* untuk menjalankan *GUI* dari *QT*. Rancangan perangkat lunak akuisisi data detektor

gamma buatan Rusia ini memanfaatkan editor *qtdesigner* dari *PyQT* untuk membuat antarmuka berbasis grafik dan menggunakan bahasa pemrograman python sebagai program yang berada dibalik antarmuka pengguna berbasis grafik tersebut.

3. TATA KERJA

3.1. Alat dan Bahan

Peralatan yang digunakan dalam merancang perangkat lunak akuisisi data detektor gamma yaitu:

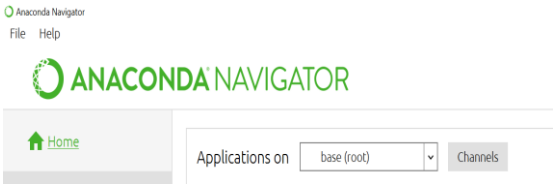
- Sebuah komputer laptop Intel CORE i7, platform windows 10, dengan kapasistas RAM sebesar 8 GB.
- Sebuah sistem detekeksi monitor lingkungan yang berisi detektor Geiger Muller dan detektor pengedip cahaya yang terbuat dari CsI(TI) dengan protokol komunikasi berbasis *Modbus* RTU yang dilewatkan pada TCP/IP.
- Software python Anaconda 3 dengan editor Spyder
- Software interface python QT, *PyQT5*.

Bahan yang digunakan terdiri dari:

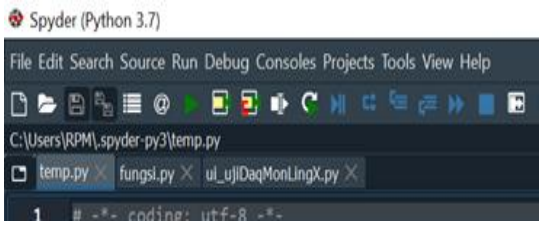
- Sebuah kabel LAN RJ45.
- Sebuah Konvertor RJ45 ke USB.

3.2. Pola Kerja

✓ Pertama, bahasa pemrograman anaconda (Gambar 2, editor spyder (Gambar 3), dan command prompt anaconda (Gambar 4) telah terpasang di laptop.

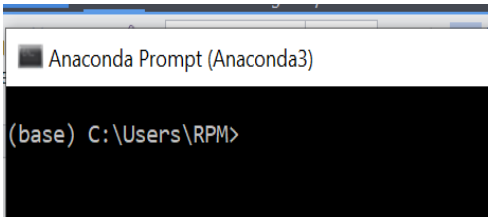


Gambar 2. Tampilan Anaconda 3

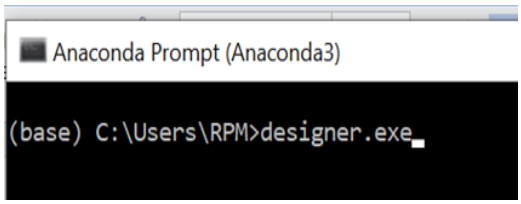


Gambar 3. Spyder, editor Anaconda

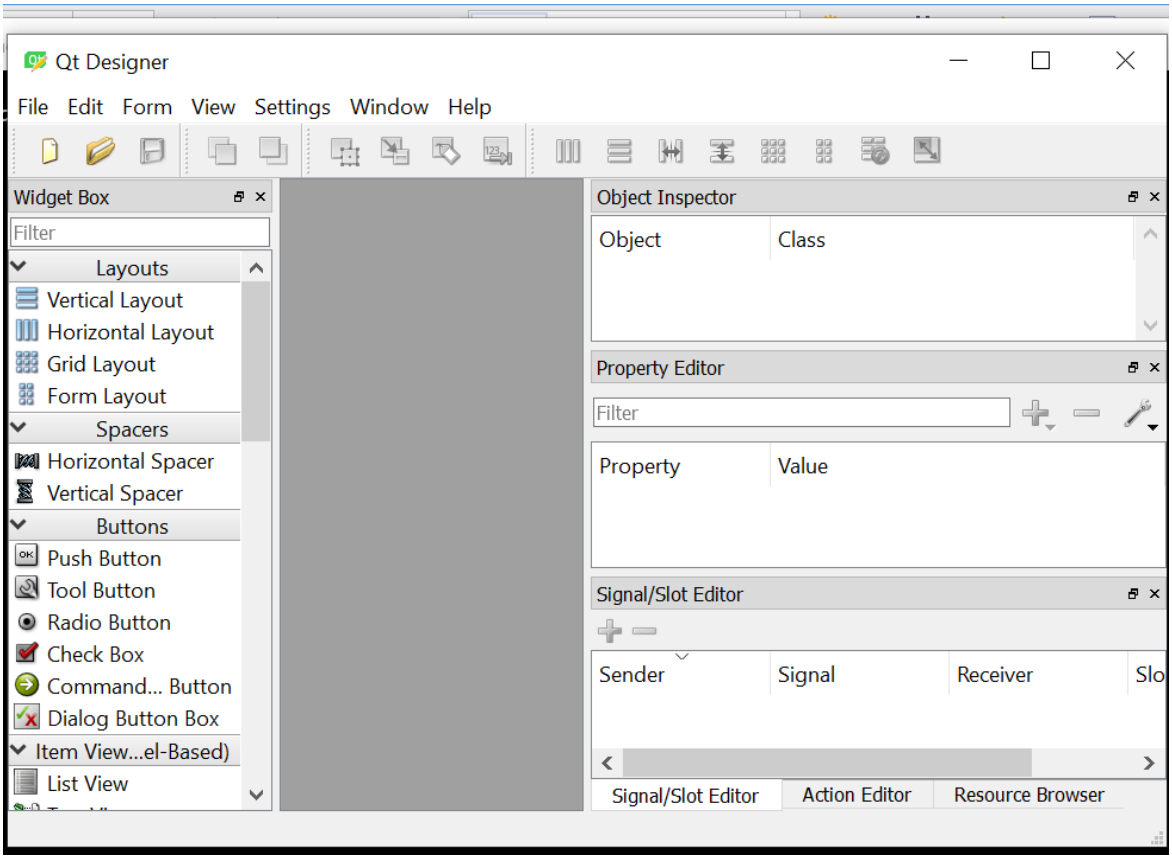
✓ Jika *PyQT5* telah terinstal, maka qtdesigner dijalankan di *command prompt anaconda* dengan mengetikkan designer.exe (Gambar 5) Tampilan editor GUI designer dari *PyQT5* dapat dilihat pada Gambar 6.



Gambar 4. Command prompt dari Anaconda



Gambar 5. Perintah qtdesigner versi PyQT5



Gambar 6. Tampilan GUI qtdesigner versi *PyQT*5

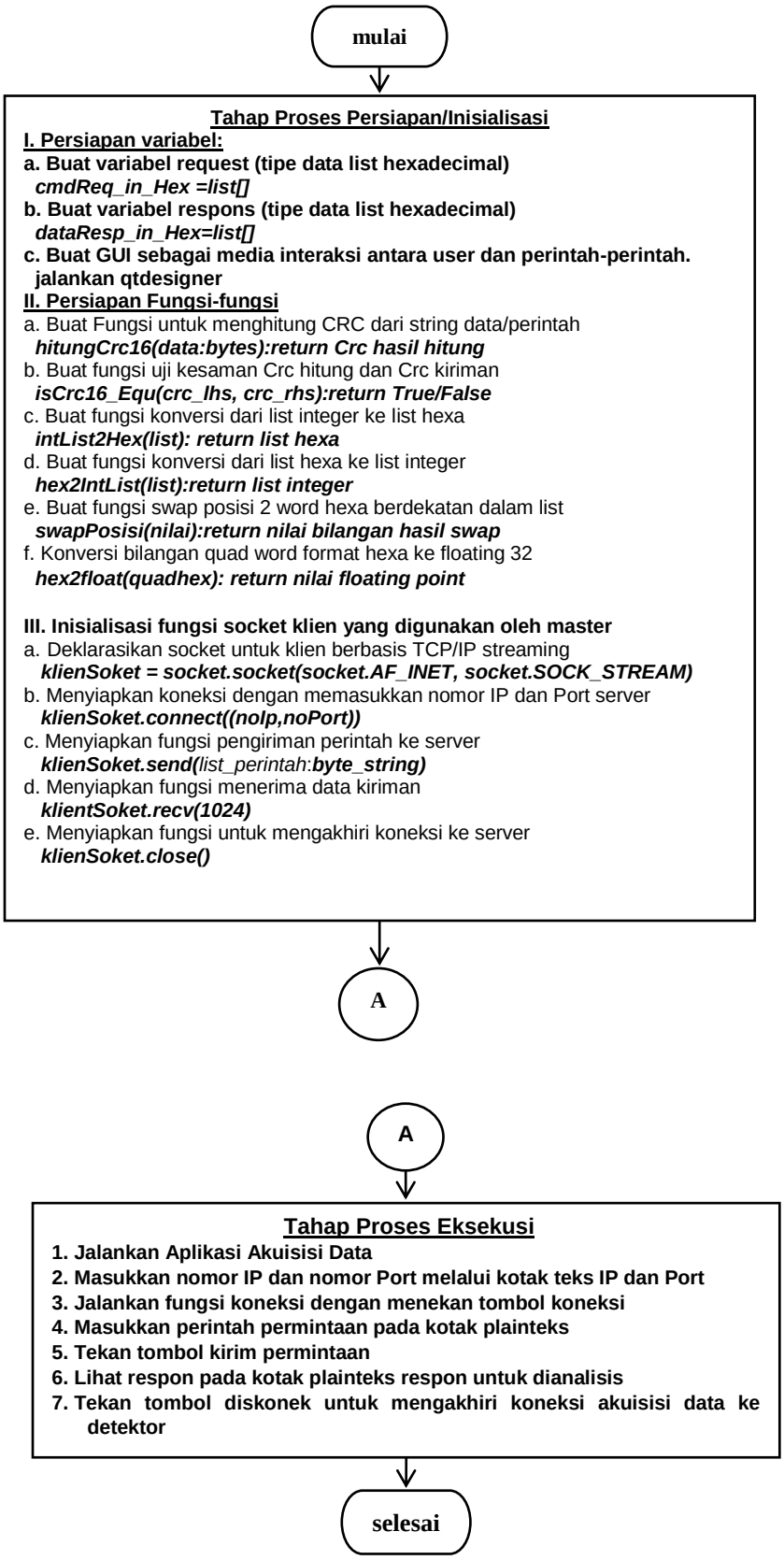
- ✓ Selanjutnya, merancang tampilan windows berbasis *GUI QT designer*.
- ✓ Tampilan windows kemudian dikompilasi dengan perintah *pyuic5* di command prompt *anaconda3* (contoh *pyuic5* winMonLing.ui -o winMonLing.py).
- ✓ Terakhir, membuat koneksi hasil kompilasi *GUI* dengan bahasa pemrograman *python*.

3.3. Perancangan

Metodologi perancangan disajikan dalam bentuk flowchart (Gambar 7). Metodologi perancangan dibagi ke dalam dua tahap proses, yaitu tahap yang disebut proses persiapan atau inisialisasi, dan tahap yang disebut proses eksekusi.

Pada tahap persiapan, dilakukan pembuatan variabel-variabel untuk menyimpan perintah permintaan dan respon, pembuatan fungsi-fungsi untuk olah data, membuat perintah-perintah koneksi jaringan menggunakan socket, perintah pengirim data, perintah menerima data, serta perintah untuk memutus komunikasi.

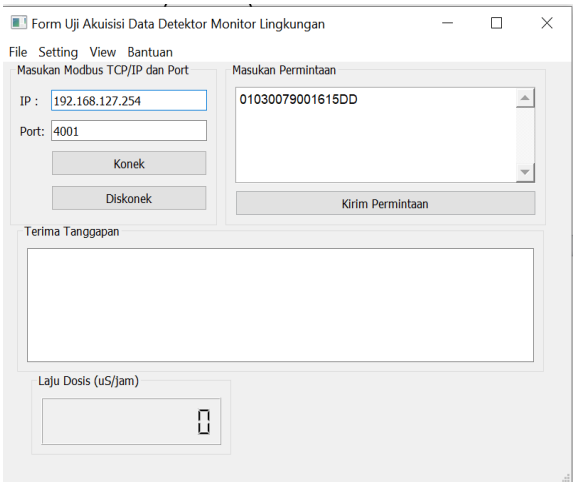
Pada tahap eksekusi, rancangan perangkat lunak akusisi data terlebih dahulu dihubungkan dengan slave, kemudian diikuti dengan memasukkan perintah permintaan. Jika komunikasi berjalan normal, maka dalam kurung waktu 1 detik, perangkat lunak tersebut akan menerima respon dari *slave*.



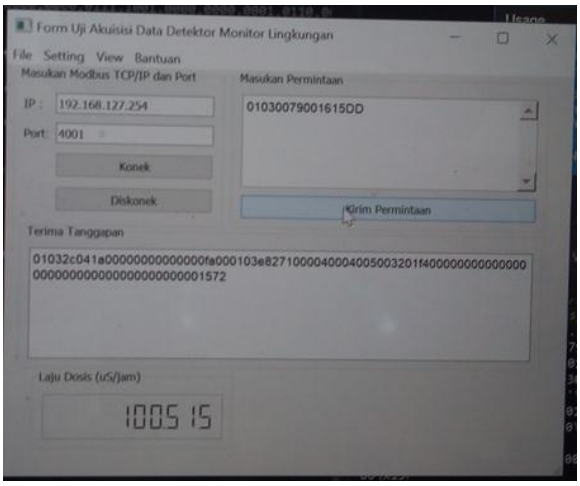
Gambar 7. Metodologi Perancangan.

4. HASIL DAN PEMBAHASAN

Hasil rancangan perangkat lunak akuisisi data dengan *PyQT*5; untuk menguji atau menganalisis format perintah dan respon data jawaban yang berasal dari sistem deteksi radiasi gamma (*slave*) berbasis protokol *modbus TCP/IP*; dapat dilihat dalam Gambar 8.



Gambar 8. *Request* dari komputer *master.*



Gambar 9. Respon dari sensor/*slave.*

Ketika perangkat lunak akuisisi data dijalankan, perangkat lunak tersebut akan menampilkan *Graphic User Interface* (GUI) seperti diperlihatkan Gambar 8. Untuk melakukan koneksi dengan sistem deteksi gamma (slave), langkah pertama yang harus dilakukan adalah dengan memasukkan nomor IP dan Port dari *slave* ke dalam kotak masukan IP dan Port masing-masing. Nomor IP dan Port slave tersebut adalah “192.168.127.254” dan “4001”. Setelah nomor IP dan Port dimasukkan, langkah selanjutnya adalah menekan tombol “Konek” pada perangkat lunak tersebut untuk melakukan sambungan komunikasi data dengan slave.

Perintah permintaan data dapat dimasukkan ke kotak editor teks dibawah label yang bertuliskan Masukan Permintaan bila dalam jangka waktu 1 detik tidak muncul pesan kesalahan yang berhubungan dengan koneksi. Dalam Gambar 8, contoh perintah permintaan yang akan dikirim ke modul sistem deteksi gamma(slave) adalah “01030079001615DD”. Perintah tersebut merupakan suatu untaian bilangan berformat heksadesimal (basis 16). Satu byte pertama dari perintah tersebut; yakni 0x01; menyatakan kode alamat salah satu detektor pada modul sistem deteksi berbasis protokol *modbus* tersebut. Alamat tersebut adalah milik detektor sintilator CsI(TI). Satu byte berikutnya; yaitu 0x03; merupakan perintah untuk membaca isi register *holding* yang merupakan tempat menyimpan data detektor CsI(TI). Dua byte setelah byte perintah baca ;yaitu 0x0079; adalah nilai yang merepresentasikan titik acuan alamat dari register *holding* pertama yang akan dibaca datanya. Alamat yang dimaksud adalah 0x007A. Selanjutnya, dua byte berikutnya lagi, yaitu 0x0016; merupakan nilai yang menyatakan jumlah register yang akan dibaca datanya. Register tersebut berawal dari register dengan alamat 0x007A. Nilai 0x0016 ekivalen dengan angka 22 dalam bilangan basis desimal. Ini berarti, komputer *master* meminta data dari 22 register yang ada di *slave*. Jika setiap register memiliki kapasistas penyimpanan 2 byte, maka jumlah data yang akan dikembalikan oleh *slave* adalah sebanyak 44 byte.

Respon *slave* terhadap perintah permintaan diperlihatkan dalam Gambar 9. Respon *slave* tersebut adalah dengan mengirim untaian bilangan heksadesimal “01 03 2c 041a 0000 0000 0000 00fa 0001 03e8 2710 0004 0004 0050 0320 1f40 0000 0000 0000 0000 0000 0000 0000 0000 0000 1572” ke *master*. Respon tersebut diawali oleh nilai 0x01 yang merepresentasikan alamat detektor di modul sistem deteksi yang melakukan

respon. Nilai berikutnya; yakni 0x03; menyatakan bentuk perintah permintaan yang dikirim kembali oleh *slave* sebagai salah satu bentuk verifikasi. Nilai berikutnya, yakni 0x2C atau 44 dalam bilangan desimal menyatakan jumlah untaian byte data yang mengiringi nilai tersebut adalah sebanyak 44 byte. Jumlah byte data tersebut merupakan jumlah byte data dari sekumpulan register dengan kapasitas penyimpanan masing-masing sebesar 2 byte. Dengan demikian jumlah byte data tersebut berasal dari 22 register *holding* yang ada di *slave* dan dimulai dari register pada alamat 0x007A. Selanjutnya, dua byte terakhir dari untaian data tersebut merupakan nilai CRC_16. Nilai ini digunakan untuk memvalidasi konsistensi data antara yang diminta dan yang terkirim. Untaian data yang divalidasi adalah seluruh untaian yang diterima selain nilai CRC_16 tersebut.

Untaian nilai-nilai yang dikirim balik oleh *slave* sebagai respon dari perintah *request* dari komputer *master* hingga saat ini masih dianalisis lebih lanjut dan tidak masuk dalam lingkup pembahasan paper ini.

5. KESIMPULAN DAN SARAN

A. Kesimpulan

1. Telah dilakukan perancangan perangkat lunak akuisisi data untuk sistem deteksi gamma yang berbasis pada protokol *modbus RTU* yang dilewatkan pada protokol internet *TCP/IP*. Rancangan perangkat lunak tersebut menggunakan *PyQT5* untuk menghubungkan bahasa pemrograman *Python* yang berbasis teks dan *QT* yang berbasis GUI.
2. Perangkat lunak hasil rancangan tersebut berfungsi untuk menganalisis bentuk-bentuk perintah *request* ke modul sistem deteksi gamma RosRoa dan bentuk respon atas perintah tersebut.

B. Saran

Rancangan perangkat lunak akuisisi data ini dapat dikembangkan lebih lanjut untuk mengatur parameter modul detektor dan menampilkan semua informasi yang berhubungan dengan data-data yang ada pada modul tersebut.

7. DAFTAR PUSTAKA

[1] Anonim., October 24, 2006, *Modbus Messaging on TCP/IP Implementation Guide V1.0b.,*Modbus Organization, Inc*.,* http://www.modbus.org/ docs/ Modbus_Messaging_implementation_Guide_V1_0b.pdf , Diunduh: 20-4 -2020.

[2] Anonim., April 26, 2012, *Modbus Application Protocol Specification V1.1b3,* Modbus Organization,Inc.,http://www.modbus.org/docs/Modbus_Application_Protocol_V1_1b3.pdf, Diunduh: 29 April 2020.

[3] Sourangsu Banerji*.,* February, 2013, *Study and Development of a Data Acquisition & Control (DAQ) System Using TCP/Modbus Protocol*., Department of Electronics & Communication Engineering, RCC-Institute of Information Technology, Under West Bengal University of Technology.

[4] Anonim, *TPU2000/2000R SERIAL MODBUS/MODBUS PLUS/MODBUS TCP/IP AUTOMATION*, TECHNICAL GUIDE TG 7.11.1.7-61 Version 1.1, 5/0.

[5] Vertiv., 2018, *Libert IntelliSlot Modbus and BACnet Protocols*. Reference Guide Modbus RTU and TCP, BACnet MSTP and IP Protocols.

[6] Lydia Parziale, David T. Britt, Chuck Davis, Jason Forrester, Wei Liu, Carolyn Matthews, dan Nicolas Rosselot., Desember 2006. *TCP Tutorial and Technical Overview*., International Business Machine Corporation, 8th Edition.

[7] Libor Dostálek, Alena Kabelová., *Understanding TCP/IP, A Clear and Comprehensive Guide to TCP/IP Protocols*., Packt Publishing Ltd. 32 Lincoln Road, Olton, Birmingham, B276PA, UK, 206. ISBN 1-904811-71-X.
[8] Behrouz A. Forouzan., 2010., *TCP/IP Protocol suite*. McGraw-Hill Forouzan Networking Series, 4th Edition., ISBN 978-0-07-337604-2.
[9] Bobbi Sandberg., 2015., *Networking. The Complete Reference*. Mc Graw Hill, ISBN: 978-0-07-182765-2
[10] Mark Summerfield., 2018., *Rapid GUI Programming with Python and QT, The Definitive Guide to PyQT Programming*. Prentice Hall, Open Source Development Series. Pearson Education, Inc. ISBN 978-0-13-235418-9.
[11] Anonim., 2015., *PyQT Python Binding, Tutorialspoint Simply Easy Learning*. Tutorials Point (I) Pvt. Ltd... http://www.tutorialspoint.com. Diunduh: 20-April 2020.
[12] Andrew Steele., March 1, 2016., *PyQt5 Tutorial Documentation*., Release 1.0.

-oOo-